1 Catalog Description

Explores the main features of modern, high-level, general purpose programming languages from the user (programmer) point of view. Provides students with an opportunity to use non-imperative programming paradigms, such as object-oriented, functional, and logical, and to learn how specific features of such languages can be used efficiently in solving programming problems. Prereq: CS619.

2 Attributes

- This course is required for CS majors. The minimum passing grade is D−.

3 Outcomes

- methodologies of software development: interfaces, modules, data encapsulation, information hiding, code reuse, testing, documentation, object-oriented design patterns, functional design patterns.
- fundamental computer science algorithms: recursive programming, dynamic programming, concurrency.
- concepts of programming languages: types, parameters, functions, mutation
- advanced topics: reification, threads, actors, serialization, client-server paradigm

4 Topics

- Basic syntax and semantics:
  - values, expressions, primitive types, variables
  - conditionals, loops, exceptions, iteration, pattern-matching
  - objects, methods, functions, parameters, types
  - infix operators, string interpolation
- Intermediate syntax and semantics:
  - modules, packages, imports, collections
  - annotations, assertions
  - generics, type inference
  - variable-length arguments, default values, pass-by-name arguments
• **Advanced syntax and semantics:**
  – implicit parameters, implicit conversions
  – user-defined pattern-matching
  – type bounds, covariance and contravariance
  – domain specific languages
  – self-types and collections design

• **Object-oriented programming:**
  – classes, objects, singletons
  – interfaces, traits, inheritance, overriding, abstract members
  – polymorphism, dynamic binding, mixins, nominal and structural subtyping
  – equality, serialization, mutability

• **Functional programming:**
  – recursion, tail recursion, memoization, dynamic programming
  – first-class functions, function literals, higher-order functions, currying, closures
  – lazy evaluation
  – monads

• **Concurrent and distributed programming:**
  – threads, synchronization
  – actors
  – sockets, servers

5 **Evaluation**

Six programming assignments (60%), one mid-term exam (20%) and one final exam (20%).

6 **Textbooks**

**Required:**


**Reference:**